Содержание

1. Лабораторная работа 6. Исследование стойкости шифрования паролей пользователей

1.1. Информация по стенду

- Учетные записи

```
sysadmin:netlab123
root:netlab123
```

1.2. Инструменты злоумышленника для атак на систему аутентификации

- Для восстановления паролей пользователей из перехваченных по сети хэш-образов или хэш-образов полученных путем доступа к базе паролей пользователей могут использоваться следующие основные инструменты
  - **John the Ripper (JtR)**
  - **hashcat**
  - множество других, как правило узкоспециализированных (для WiFi, SQL и т.п.)

1.3. John the Ripper (JtR)

- Один из самых старых и известных инструментов

```
http://www.openwall.com/john/
```

- Предназначен для вскрытия различных типов хэшей (перехваченных или по базе)
- Режимы перебора
  - **bruteforce** - метод грубой силы
  - **подбор по словарю**
  - **гибридный**
- Умеет автоматически определять тип используемого хэширования

```
$ apt update && apt install john
```

Установка в Debian Linux

```
$ john
```

Запуск

```
$ john --list=opencl-devices
```

умеет работать через OpenCL (CUDA, etc)

1.3.1. Подготовка базы паролей для JtR

- JtR умеет работать с базой данных учетных записей пользователей до ее разделения на

```
$ unshadow /etc/passwd /etc/shadow > passwd.txt
```

Требует привилегий суперпользователя для доступа к /etc/shadow

1.3.2. JtR: Single crack mode

- **Single crack mode** - поиск пароля по косвенным уликам.

- Первыми кандидатами в пароли становятся имя пользователя, «GECOS», «Full Name» - поля его учётной записи и название домашней директории пользователя.
- А также используются правила (гибридная атака) для полученных данных, и возможных паролей становиться больше
- Самый быстрый способ

```
$ john --single passwd.txt
```

1.3.3. JtR: Worldlist

- Подбор пароля по словарю
  - зависит от размера словаря
  - базовый поставляется вместе с утилитой
- У режима wordlist есть «подрежим»: wordlist с правилами (rulets).
  - Это гибридный вид подбора пароля.
  - Набор применяемых правил можно изменять и дополнять своими

```
$ john --wordlist=dictfile passwd.txt
```

dictfile - файл словаря

1.3.4. JtR: Incremental

- грубый перебор, т.е. брутфорс.
- Настройки перебора хранятся в файле конфигурации.
- При переборе, программа равномерно распределяет частоту инкремента по длине пароля и набору символов

1.3.5. JtR: External

- даёт возможность применять «фильтры», описанные в файле конфигурации на языке Си с использованием четырёх callback-функций.
- С помощью этого можно написать свой алгоритм перебора

1.3.6. JtR: Johnny GUI

- Поставляется отдельно, в дистрибутивах общего назначения может отсутствовать

1.4. hashcat

- Установка (Debian Linux)

```
$ apt update && apt install hashcat
```

- Тестирование скорости перебора паролей по базе в зависимости от алгоритма кэширования

```
$ hashcat -b
```

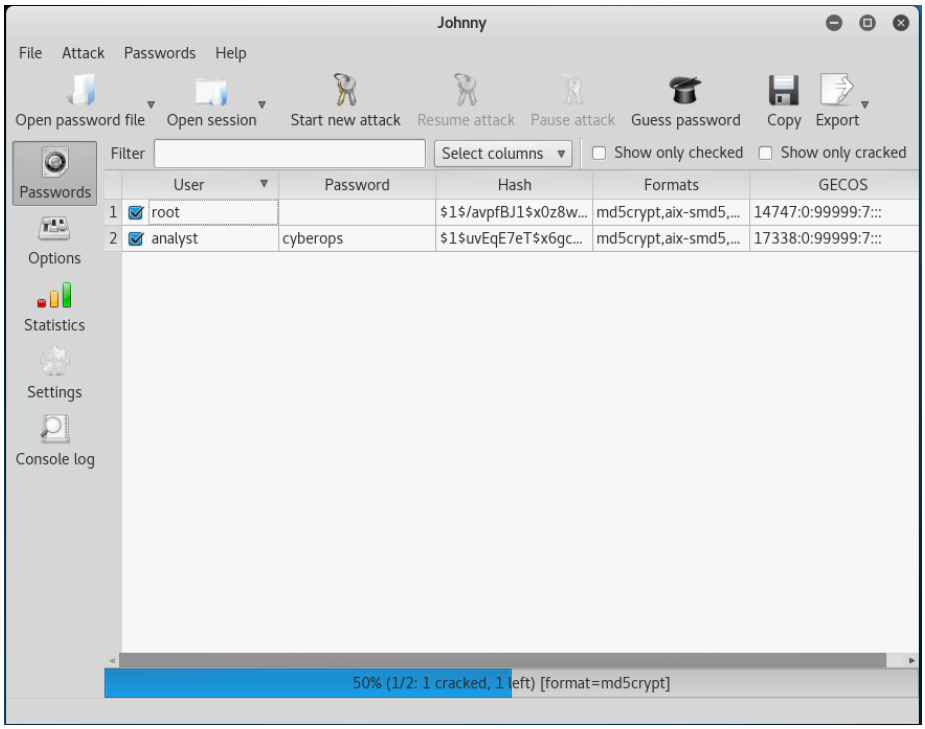
Johnny
File
Attack
Passwords
Help
Open password file
Open session
Start new attack
Resume attack
Pause attack
Guess password
Copy
Export
Filter
Select columns
Show only checked
Show only cracked
Passwords
Options
Statistics
Settings
Console log
User
Password
Hash
Formats
GECOS
1
root
1/avpfBJ1$x0z8w...
md5crypt,aix-smd5,...
14747:0:99999:7:::
2
analyst
cyberops
1uvEqE7eT$x6gc...
md5crypt,aix-smd5,...
17338:0:99999:7:::
50% (1/2: 1 cracked, 1 left) [format=md5crypt]

Рис. 1: GUI JtR

1.4.1. hashcat: Режимы атак (Attack mode)

- **Straight (-a 0)** - обычная атака, берет слова из словаря и проверяет их;
- **Combination (-a 1)** - комбинирует слова из словаря в разные комбинации;
- **Toggle-Case (-a 2)** - по очереди пробует разный регистр букв для каждого символа слова;
- **Brute-force (-a 3)** - атака простым перебором на основе маски или символов;
- **Permutation (-a 4)** - при этом типе атаки программа берет слова из словаря и меняет в них буквы местами для получения разных комбинаций;
- **Table-Lookup (-a 5)** - Табличная атака, берется одно слово и словаря, а затем на его основе создаются варианты из таблицы. Каждый символ из таблицы будет заменен на набор прописанных вариантов;
- **Hybrid Wordlist + Mask (-a 6)** - слово из словаря, потом перебор на основе маски
- **Hybrid Mask + Wordlist (-a 7)** - слово из словаря, потом маска
- **Prince (-a 8)** - новый вид атаки перебора, которая работает быстрее, обычной.

  В конкретной, не слишком свежей версии **hashcat** могут быть доступны не все режимы.

1.4.2. hashcat: словари

- Типовой словарь для hashcat (130 Mb)

```
$ wget http://scrapmaker.com/data/wordlists/dictionaries/rockyou.txt
$ head rockyou.txt
123456
12345
123456789
password
iloveyou
princess
1234567
rockyou
12345678
abc123
```

  Содержит как сами слова “из словаря”, так и наиболее типовые комбинации, используемые для паролей, типа **qwerty**, **123456**, так и разного рода вариации со словарными словами, например **cucumber123**, **13cucumber13**, **seacucumber**

- Проверка наличие слова/части слова в словаре

```
$ grep barber rockyou.txt
```

1.4.3. Задание 1. Установка john, hashcat и словаря для атак перебора

1. Выполните установку утилит **john** и **hashcat**
2. Скачайте типовой словарь rockyou, воспользовавшись ссылкой, приведенной выше
3. Протестируйте скорость работы hashcat, сравните его скорость для различных хэш-алгоритмов

1.4.4. hashcat: создаем файл хэш-образов паролей

- Для тестирования скорости перебора разных паролей по базе удобно использовать самостоятельно сгенерированный файл хэшей паролей, например вот такой

```
$ echo -n "password" | md5sum | cut -f1 -d' ' > hashes.md5
$ echo -n "secret" | md5sum | cut -f1 -d' ' >> hashes.md5
$ echo -n "qwerty" | md5sum | cut -f1 -d' ' >> hashes.md5
$ cat hashes.md5
5f4dcc3b5aa765d61d8327deb882cf99
5ebe2294ecd0e0f08eab7690d2a6ee69
d8578edf8458ce06fbc5bb76a58c5ca4
```

1.4.5. hashcat: стандартная атака по словарю (-a 0)

- Проведение атаки по словарю средствами **hashcat**
  - режим **0**

```
$ hashcat
Usage: hashcat [options] hashfile [mask|wordfiles|directories]
```

Результат будет успешным только в том случае, если искомый пароль содержится в словаре

```
$ grep cucumber123 rockyou.txt
cucumber12345
cucumber123
```

- Вызов **hashcat** для раскрытия паролей в ранее созданном файле хэшей

```
$ hashcat -O -m 0 -a 0 hashes.md5 rockyou.txt
. . .
5f4dcc3b5aa765d61d8327deb882cf99:password
d8578edf8458ce06fbc5bb76a58c5ca4:qwerty
5ebe2294ecd0e0f08eab7690d2a6ee69:secret
. . .
Status...........: Cracked
. . .
Speed.#1.........:  726.8 kH/s (0.94ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
Recovered........: 3/3 (100.00%) Digests
. . .
```

- **-m 0** - режим перебора хешей md5
- **-a 0** - режим атаки по словарю
- **-O** - (большая буква O) стандартная оптимизация
- **hashes.md5** - сгенерированный ранее файл MD5 хэшей
- **rockyou** - файл словаря
- Результат кроме вывода на экран будет записан в файл **.hashcat/hashcat.potfile**

```
$ cat .hashcat/hashcat.potfile
5f4dcc3b5aa765d61d8327deb882cf99:password
d8578edf8458ce06fbc5bb76a58c5ca4:qwerty
5ebe2294ecd0e0f08eab7690d2a6ee69:secret
```

Файл, содержащий восстановленные пароли (будет перезаписываться при следующих успешных сессиях hashcat)

- Для долгосрочного сохранения результата можно указывать куда записывать восстановленные пароли
- **-o file** (маленькая буква o) - записать результат в файл **file**

```
$ hashcat -O -m 0 -a 0 -o result.md5 hashes.md5 rockyou.txt
```

- Альтернативно, можно указывать хэш прямо в строке

```
$ echo -n "cucumber12345" | md5sum
7b641ec4448fbb72edc8a58aad47311f  -
$ hashcat -O -m 0 -a 0 7b641ec4448fbb72edc8a58aad47311f rockyou.txt
. . .
7b641ec4448fbb72edc8a58aad47311f:cucumber12345
. . .
Status...........: Cracked
. . .
Speed.#1.........:  1203.7 kH/s (0.72ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests
```

Получаем MD5-хэш пароля и передаем его команде hashcat

1.4.6. Задание 2. Восстановление паролей из хэш-образов MD5 по словарю

1. Убедитесь в **присутствии** в используемом словаре следующих слов:
   - **17barb1e**
   - **suriken77**
   - **007potatoice**
2. Сгенерируйте хэши MD5 для этих слов и сохраните их в файл, как было показано в примерах выше
3. При помощи **hashcat** выполните восстановление паролей из образов.
4. Результат восстановления сохраните в файл **result.md5**

- Файл должен содержать 3 строчки (по количеству слов), в каждой строчке присутствовать значения хэша и пароля, разделенные двоеточием.

5. Полученный файл **result.md5** является одним из результатов выполнения работы.

1.4.7. hashcat: использование других хэш-функций

- Меняя значения параметра **-m** можно производить восстановление хэш-образов, посчитанных по другим алгоритмам.
- Например, **-m 1400** - восстановление хэш-образа SHA-256
- Все поддерживаемые хэш-образы и соответствующие им значения параметра **-m** можно посмотреть в справке по команде **hashcat** в разделе **Hash types**
- **Примечание.** Для создания хэш-образов кроме программы **md5sum**, есть аналогичные программы **sha1sum,sha224sum,sha256sum,sha384sum,sha512sum** - для семейства хэш-функций SHA/SHA2 а так же ряд других.

```
$ find /usr/bin -type f -name '*sum'
```

Поиск подобных подпрограмм (по характерному суффиксу в имени)

1.4.8. Задание 3. Восстановление паролей из хэш-образов SHA512 по словарю

1. Возьмите 5 случайных парольных комбинаций, присутствующих в используемом словаре (проверьте) , но не используемых ранее в примерах и заданиях.
2. Сгенерируйте хэши SHA512 для этих слов и сохраните их в файл, как было показано в примерах выше. Команду генерации кэшей найдите самостоятельно.
3. При помощи **hashcat** выполните восстановление паролей из образов. Значение параметра **-m** для восстановления образов SHA512 определите по man-странице утилиты **hashcat**
4. Результат восстановления сохраните в файл **result.sha512**

- Файл должен содержать 5 строчек (по количеству слов), в каждой строчке присутствовать значения хэша и пароля, разделенные двоеточием.

5. Полученный файл **result.sha512** является одним из результатов выполнения работы.

1.4.9. hashcat: перебор по словарю с заменой букв местами (-a 4)

- Часто пароли создают на базе типовых слов, меняя буквы местами

  Например, словарное слово **seacucmber** путем перестановки символов можно превратить в **saecucubmer**

```
$ grep saecucubmer rockyou.txt
```

Такого слова действительно нет в словаре

- Атака по словарю на хэш, полученный из такого пароля, результата не даст

```
$ echo -n 'saecucubmer' | md5sum
3bf498733285c5d09262703e1257acc7  -
$ hashcat -O -m 0 -a 0 3bf498733285c5d09262703e1257acc7 rockyou.txt
. . .
Status...........: Exhausted
. . .
```

- Восстановление пароля с использованием 4го режима атаки

```
$ hashcat -O -m 0 -a 4  3bf498733285c5d09262703e1257acc7 rockyou.txt
```

Permutation - замена букв местами. Если такой режим поддерживается используемой версией **hashcat** можно относительно быстро получить результат.

1.4.10. hashcat: наборы символов для атак перебора

- Для режимов атак, полагающихся не на наличие пароля в словаре, а на перебор, **hashcat** позволяет задавать множества символов, среди которых будет производитьсмя перебор.

```
?l = abcdefghijklmnopqrstuvwxyz;
?u = ABCDEFGHIJKLMNOPQRSTUVWXYZ;
?d = 0123456789;
?s = !"#$%&'()*+,-./:;<=>?@[]^_`{|}~;
?a = ?l?u?d?s - любой символ;
?b = 0x00 - 0xff.
```

1.4.11. hashcat: атака методом перебора (-a 3)

- Вызов **hashcat** для перебора 5ти символьных паролей

```
$ hashcat -O -m 0 -a 3  hashes ?u?u?u?u?u
```

- **-a 3** режим перебора (brute force)
- **?u?u?u?u?u** - 5 заглавных символов
- Для проверки работы **hashcat** в режиме **brute force** (атака методом перебора) для начала сгенерируем случайные 4ех-символьные пароли с использованием генератора псевдослучайных чисел

```
$ dd if=/dev/urandom bs=1 count=20 | base64
20+0 записей получено
20+0 записей отправлено
gMcZHYxl8LTK8IqcI2fBzwsWLLY=
20 байт скопировано, 0,00105404 s, 19,0 kB/s
```

Генерируем последовательность из 20 псевдослучайных чисел, преобразуем их в читабельный вид для корректного отображения на терминале (получится больше чем 20 читаемых символов).

```
$ echo -n "gMcZ" | md5sum | cut -d' ' -f1 > random-hashes.md5
$ echo -n "HYxl" | md5sum | cut -d' ' -f1 >> random-hashes.md5
$ echo -n "8LTK" | md5sum | cut -d' ' -f1 >> random-hashes.md5
```

Разрезаем псевдослучайную строку **gMcZHYxl8LTK8IqcI2fBzwsWLLY=** по 4 символа и создаем файл хэшей для получившихся паролей

```
$ hashcat -O -m 0 -a 3 -o random-result.md5 random-hashes.md5 ?a?a?a?a
```

Вызываем **hashcat** для восстановления паролей методом перебора

```
$ cat random-result.md5
8b739c56d9533e991feb504be870961d:HYxl
af504ae58e29881a3dbef9057b48bbdc:8LTK
a3d268e0dc6314e637f3eff132bbb9e8:gMcZ
```

Полученный результат

1.4.12. Задание 4. Восстановление паролей из хэш-образов MD5 методом перебора

1. Создайте с использованием генератора псевдослучайных чисел 4 парольные комбинации, длиной по 5 символов каждая, не используемых ранее в примерах и заданиях.
2. Проверьте их отсутствие в используемом словаре.
3. Сгенерируйте хэши MD5 для этих слов и сохраните их в файл, по аналогии с предыдущими заданиями.
4. При помощи **hashcat** выполните восстановление паролей из образов.
5. Результат восстановления сохраните в файл **result-random.md5**

- Файл должен содержать 4 строчки (по количеству слов), в каждой строчке присутствовать значения хэша и пароля, разделенные двоеточием.

6. Полученный файл **result-random.md5** является одним из результатов выполнения работы.

1.5. Итоговое задание. Загрузка результата выполнения лабораторной работы

- В качестве результата выполнения лабораторной работы необходимо загрузить в Moodle следующие файлы:
  - Файл **result.md5**, полученный в задании 2
  - Файл **result.sha512**, полученный в задании 3
  - Файл **result-random.md5**, полученный в задании 4